Transformación del modelo de análisis al modelo de diseño utilizando QVT

Ariel Arsaute, Marcela Daniele, Mariana Frutos, Ariel Gonzalez, Paola Martellotto,
Daniel Riesco [(1)], Marcelo Uva, Fabio Zorzan.
Departamento de Computación, Facultad de Ciencias Exactas, Físico-Químicas y Naturales,
Universidad Nacional de Río Cuarto.
Ruta 36 Km. 601 –CP 5800 - Río Cuarto – Córdoba - Argentina Tel. (0358) 4676235
{aarsaute, marcela, mfrutos, agonzalez, paola, uva, fzorzan}@dc.exa.unrc.edu.ar

[(1)] Universidad Nacional de San Luis
Ejercito de Los Andes 950 - CP D5700HHW - San Luis – Argentina – Tel (2652) 424027 int 251
driesco@unsl.edu.ar

Resumen

Este trabajo realiza un aporte tendiente a la mejora del proceso de desarrollo de software, siguiendo la filosofía de Model Driven Architecture (MDA). Dentro de esta línea de investigación en un trabajo previo se definió la transformación del modelo de CU de sistema al modelo de análisis, para lo cual se tuvo que definir un metamodelo correspondiente a Plantillas que se utilizan para la especificación de requisitos de CU de sistemas, a partir de esto se especificó una serie reglas de transformación Query/View/Trasmormation (QVT) que establecieron una correspondencia entre los modelos correspondientes a las etapas de captura de requisitos y análisis.

El modelo de análisis involucrado en la transformación del trabajo previo es un modelo UML, que en este trabajo propuesto lo tomaremos cómo modelo fuente, y el modelo objetivo de las transformación QVT propuesta será un diagrama de clases de diseño, que es también un modelo UML.

El objetivo final es la realización de aportes tendientes a lograr un producto de software siguiendo los principios de abstracción, automatización y estandarización establecidos por MDA, de manera totalmente automática, o bien automatizando el proceso de transformación lo mayormente posible.

Palabras clave: MDA, QVT, Proceso Unificado, UML.

Contexto

La línea de investigación presentada en este trabajo se desarrolla en el marco del proyecto “Ingeniería de Software: Automatización de Procesos de Desarrollo de Software”, perteneciente a los Proyectos y Programas de Investigación (PPI) de la secretaría de Ciencia y Técnica de la Universidad Nacional de Río Cuarto.

Introducción

La ingeniería de Software implementa un enfoque sistemático, disciplinado y cuantificable para el desarrollo, operación y mantenimiento del software. Las metodologías de desarrollo junto con las herramientas de software se han adoptado con éxito en un amplio espectro de aplicaciones industriales. Dentro de las metodologías orientadas a objetos, el Proceso Unificado [1] define actividades y responsabilidades estableciendo quién está haciendo qué, cuándo, y cómo para construir o mejorar un producto de software. Esta metodología, divide el desarrollo del producto de software en fases utilizando UML (Unified Modeling Language) como lenguaje de modelado [2].

ARQUITECTURA DIRIGIDA POR MODELOS

La Arquitectura Dirigida por Modelos (MDA) [6,7] se ha concebido para dar soporte a la ingeniería

dirigida por modelos de los sistemas software, cuyo objetivo central es resolver el problema producido por el cambio de tecnologías junto con la integración del sistema de software y que todo esto no impliquen un alto costo.

La idea subyacente en MDA es usar modelos, de modo que las propiedades y características de los sistemas queden contenidas en un modelo abstracto independiente de los cambios producidos en la tecnología. MDA proporciona una serie de guías o patrones expresadas como modelos. MDA propone cuatro niveles de abstracción que componen la jerarquía o arquitectura de modelos. Estos son: Modelo independiente de cómputo CIM (Computation Independent Model), Modelo independiente de la plataforma PIM (Platform Independent Model), Modelos específicos de la plataforma PSM (Platform Specific Model), y la aplicación final.

Los modelos CIM describen el entorno en el que se utilizará el sistema, sin referencia directa a su implementación.

Los PIM modelan funcionalidad y estructura de un sistema de información sin considerar detalles

tecnológicos de la plataforma en la cual se implementará el sistema.

Los PSM describen los modelos específicos de plataforma, concretamente de la plataforma tecnológica donde se ejecutará el sistema.

MDA plantea el siguiente proceso de desarrollo: de los requisitos se obtiene un modelo independiente de la plataforma (PIM), luego este modelo es transformado con la ayuda de herramientas en uno o más modelos específicos de la plataforma (PSM), y finalmente cada PSM es transformado en código. Por lo tanto, MDA incorpora la idea de transformación de modelos (PIM a PSM, PSM a código) y se necesitan herramientas para automatizar estas tareas. Estas herramientas constituyen uno de los elementos básicos de MDA.

QVT (Query / View / Transformation)

El planteamiento QVT se basa principalmente en la definición de un lenguaje para las consultas (Queries) sobre los modelos Meta Object Facility (MOF) [4], la búsqueda de un estándar para generar vistas (Views) que revelen aspectos específicos de los sistemas modelados, y finalmente, la definición de un lenguaje para la descripción de transformaciones de modelos MOF.

En este trabajo se plantea la utilización de QVT para definir transformaciones entre modelos. Estas transformaciones describen relaciones entre un metamodelo fuente F y un metamodelo objetivo O, ambos metamodelos deben estar especificados en MOF. Luego esta transformación se utiliza para obtener un modelo objetivo, el cual es una instancia del metamodelo O, a partir de un modelo fuente que es una instancia del metamodelo F. Una característica muy importante de estas transformaciones es que pueden ser bidireccionales y multidimencionales.

Líneas de investigación y desarrollo

Como se mencionó anteriormente, MDA propone basar el desarrollo de software en modelos, separando el diseño de la arquitectura y de las tecnologías de construcción, posibilitando que el diseño y la arquitectura puedan ser alterados independientemente. Una vez definidos los modelos, se definen una serie de trasformaciones que generan nuevos modelos [5], permitiendo ir de modelos más abstractos a otros más concretos. MDA define un framework para procesar y relacionar modelos.

La transformaciones entre modelos se realizan haciendo uso de herramientas automáticas, como herramientas de transformación de modelos, las cuales permiten el refinamiento de los mismos.

Nuestra propuesta consiste en tomar el modelo de análisis UML obtenido como resultado del trabajo anterior [3] el cual corresponde a una instancia del metamodelo UML, como se puede ver en la Fig.1. A partir de este modelo, definiremos reglas de transformación QVT que permitirán establecer una correspondencia entre los modelos de análisis y diseño respectivamente. De esta manera se pretende lograr un acercamiento a la automatización de la

construcción del software aplicada a aquellos procesos de desarrollo dirigidos por casos de uso.

Resultados y Objetivos

En esta etapa se trabajará directamente en la definición de reglas de transformación QVT, debido a que los metamodelos de entrada y salida son el metamodelo UML. En la Fig.1 se puede observar las metaclases UML que se utilizan para representar los modelos de análisis y diseño.

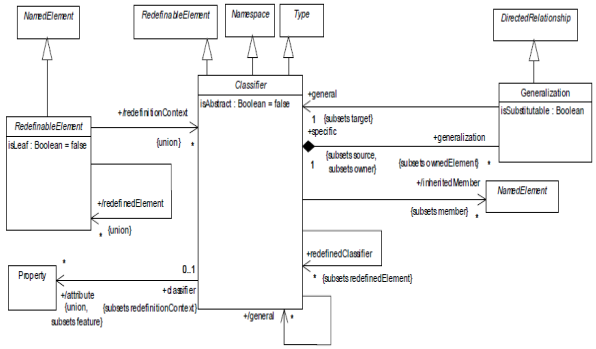

Fig. 1. Metamodelo UML 2.0 (Diagrama de Clases)

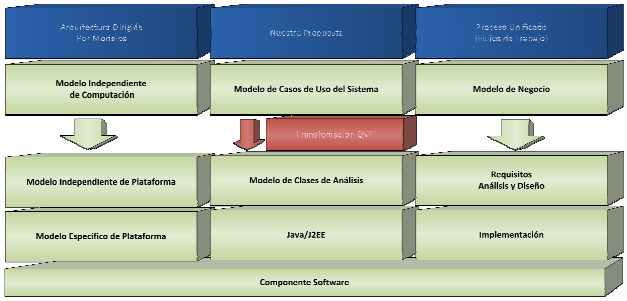

Fig. 2. Transformación modelo CUs a Modelo de Clases de Análisis

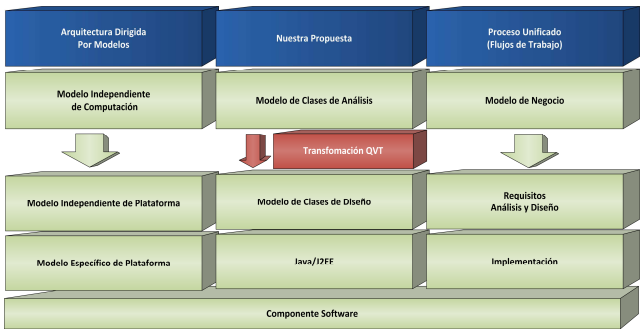

Fig. 3. Transformación Modelo Clases Análisis a Modelo de Diseño

En la Fig. 2 se puede observar la transformación definida entre el modelo de CU de sistema y el modelo de análisis. En esta futura línea de investigación se pretende utilizar el resultado de la transformación ya definida como modelo de entrada para la nueva transformación y así obtener el Modelo de clases de diseño como se puede ver en la Fig. 3.

El resultado buscado está estrechamente vinculado al logro de un acercamiento hacia la automatización de las actividades que componen el proceso de desarrollo de software, transformado modelos ya validados como lo son las Plantillas Genéricas de descripción, análisis y diseño de casos de uso, con el fin de obtener un producto de software de calidad, cumpliendo con los requerimientos iniciales.

Formación de Recursos Humanos

Durante el desarrollo de esta línea de investigación han logrado obtener el título Magister en Ingeniería de Software tres integrantes del grupo de trabajo. Otro integrante está actualmente trabajando es su tesis de Magister. Dos grupo de alumnos están realizando su trabajo final de fin de carrera de Analista en Computación, otro grupo ha logrado obtener el título de Licenciado en Ciencias de la Computación.

También, se han formado ayudantes de segunda en las asignaturas de Análisis y Diseño de Sistemas, Ingeniería de Software, Base de Datos y Proyecto.

Los temas abordados en esta línea de investigación brindan un fuerte aporte al proceso de perfeccionamiento continuo de los autores de carreras de computación en Universidades Nacionales como del exterior.

Referencias

[1] Jacobson, I. El Proceso Unificado de Desarrollo de software. Addison-Wesley, EE.UU., 2000.

[2] UML, Unified Modeling Language (UML) Resource Page. http://www.omg.org/#UML2.0

[3] Marcela Daniele, Ariel Arsaute, *et al*. "Transformación de modelos Aplicada a la definición Genérica de casos de Uso Utilizando QVT

(Query/View/Transformation) y RTG (Reglas de Transformación de Grafos)". XIII Workshop de Investigadores en Ciencias de la Computación 2011.

[4] Object Management Group, "Meta Object Facility (MOF)2.0Query/View/Transformation Specification" http://www.omg.org/docs/ptc/05-11-01.pdf.

[5] N. Debnath, F. A. Zorzan, G. Montejano and D. Riesco, "Transformation of BPMN Subprocesses Based in SPEM Using QVT", 2007 IEEE INTERNATIONAL CONFERENCE on ELECTRO/INFORMATION TECHNOLOGY, May 17-20, 2007, Marriott O Hare, Chicago, IL, USA. http://www.eit-conference.org/eit2007/

[6]. Miller, J., Mukerji, J., MDA Guide Version 1.0.1 Document number omg/2003-06-01, Disponible en: http://www.omg.com/mda, 2003.

[7] Object Management Group "Meta Object Facility (MOF) Core Specification" OMG http://www.omg.org/docs/formal/06-01-01.pdf.co@unsl.edu.ar}